Progress Report on Development of Algorithms for Low-Frequency Sonar Sensing

Lawrence Carin, Nilanjan Dasgupta and Steven Haron

Signal Innovations Group, Inc.

Durham, NC

28 November 2006

REPORT DOCUMENTATION PAGE	*Form Approved* *OMB No. 0704-0188*

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
11/29/2006	Progress Report	November 29, 2005 - November 29, 2006

4. TITLE AND SUBTITLE

Progress Report on Advanced Detection and Classification Algorithms for Acoustic-Color-Based Sonar Systems

5a. CONTRACT NUMBER

N00014-06-C-0026

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Dr. Lawrence Carin

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Signal Innovations Group
1009 Slater Road, Suite 200
Durham, NC 27703

8. PERFORMING ORGANIZATION REPORT NUMBER

CDRL A002

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Office of Naval Research
875 N. Randolph Street, Suite 1425
Arlington, VA 22203-1995

10. SPONSOR/MONITOR'S ACRONYM(S)

ONR

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

CDRL A002

12. DISTRIBUTION/AVAILABILITY STATEMENT

Statement A

13. SUPPLEMENTARY NOTES

14. ABSTRACT

This report summarizes recent progress by Signal Innovations Group (SIG) in supporting the Naval Research Laboratory (NRL) on development of a new low-frequency sonar system. SIG has the tasks of developing the algorithms and transitioning them to NRL, for use in sea tests. The discussion below provides a summary of the following items: (i) a kernel-based matching pursuits classification algorithm, (ii) life-long learning, (iii) in situ learning, and (iv) a discussion of the features used within the algorithms. Items (i) and (iv) are fully transitioned to NRL, and have been employed during sea tests. Items (ii) and (iii) are currently under development by SIG, in cooperation with NRL.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Larry Carin
					19b. TELEPHONE NUMBER *(Include area code)*
U	U	U	UU	6	(919) 323-3449

This report summarizes recent progress by Signal Innovations Group (SIG) in supporting the Naval Research Laboratory (NRL) on development of a new low-frequency sonar system. SIG has the tasks of developing the algorithms and transitioning them to NRL, for use in sea tests. The discussion below provides a summary of the following items: (i) a kernel-based matching pursuits classification algorithm, (ii) life-long learning, (iii) *in situ* learning, and (iv) a discussion of the features used within the algorithms. Items (i) and (iv) are fully transitioned to NRL, and have been employed during sea tests. Items (ii) and (iii) are currently under development by SIG, in cooperation with NRL.

A. Kernel Matching Pursuits (KMP)

Assume the feature vector is a *d*-dimensional real vector $\mathbf{x} \in \Re^d$, which we wish to map to a label $y \in \{0,1\}$; label *y*=1 may correspond to a mine, and *y*=0 to clutter. We aim to learn the optimal parameters w of the functional relationship $y = f(\mathbf{x}, \mathbf{w})$ between *d* independent feature variables x and the dependent output variable *y*. To accomplish this learning task we are provided with N labeled observations, $\{\mathbf{x}_i, y_i\}_{i=1}^N$ that are assumed to be independently and identically drawn from an unobservable underlying distribution. We are interested in learning sparse kernel machines of functional form

$$f_n(\mathbf{x}) = \sum_{i=1}^{n} w_{n,i} K(\mathbf{c}_i, \mathbf{x}) + w_{n,0} = \mathbf{w}_n^T \boldsymbol{\phi}_n(\mathbf{x}) \tag{1}$$

where $w_{n,0}$ is the bias term, $K(\cdot, \cdot)$ is a kernel function measuring the similarity between two data samples

$$\boldsymbol{\phi}_n(\cdot) = [1, K(\mathbf{c}_1, \cdot), K(\mathbf{c}_2, \cdot), ..., K(\mathbf{c}_n, \cdot)]^T \tag{2}$$

with $\{y_i\}_{i=1}^N$ the kernel-induced basis function centered at $\mathbf{c}_i$, and

$$\mathbf{w}_n = [w_{n,0}, w_{n,1}, \ w_{n,2}, \ \cdots, \ w_{n,n}]^T \tag{3}$$

are the weights that combine the basis functions in the summation, and the subscript *n* is used to denote the number of basis functions being used, with *n*<*N*. In the context of the binary classification problem consider in this section, a given x is mapped to an estimated

$y \in \{0,1\}$ as $y = \mathrm{U}[f(\boldsymbol{x}) - 0.5]$, where $\mathrm{U}(\alpha)$ is a unit step function, equal to one for $\alpha \geq 0$, and equal to zero otherwise.

The KMP implements a set of functions of the form in (3). Assume we are given a training set $\{\mathbf{x}_i, y_i\}_{i=1}^N$, where x_i is the i^{th} input and y_i its expected output. The weighted sum of squared errors between the expected output and the KMP output given in (3) is

$$e_n = (1 \big/ \textstyle\sum_{i=1}^N \beta_i) \sum_{i=1}^N \beta_i [y_i - f_n(\mathbf{x}_i)]^2 = (1 \big/ \sum_{i=1}^N \beta_i) \sum_{i=1}^N \beta_i [y_i - \mathbf{w}_n^T \boldsymbol{\phi}_n(\mathbf{x}_i)]^2 \qquad (4)$$

where β_i is a constant responsible for quantifying the importance of the *i*th training sample $(\mathbf{x}_i, y_i)$. For example, $1/\beta_i$ may represent the variance of the *i*th measurement; noisy measurements will therefore be given less importance when learning the model. In addition, if one has *a priori* knowledge that some data x_i are in some sense “better” representative of the system being modeled this can be accounted for in the parameter β_i. The unknowns in (4) are the centers $\mathbf{c}_i$ of the basis functions in $\boldsymbol{\phi}_n$, and the weights are represented by $\mathbf{w}_n$. At the moment we suppose $\mathbf{c}_i$ and consequently $\boldsymbol{\phi}_n$ are known and aim at solving for $\mathbf{w}_n$; below we address determining c_i. Then the value of $\mathbf{w}_n$ that minimizes (4) is found to be

$$\mathbf{w}_n = \mathbf{M}_n^{-1} \{\beta_i \boldsymbol{\phi}_{n,i} y_i\}_i \qquad (5)$$

where $\boldsymbol{\phi}_{n,i}$ is an abbreviation of $\boldsymbol{\phi}_n(\mathbf{x}_i)$, $\{\cdot\}_i \overset{Def}{=} \sum_{i=1}^N (\cdot)$, and

$$\mathbf{M}_n = \sum_{i=1}^N \beta_i \boldsymbol{\phi}_n(\mathbf{x}_i) \boldsymbol{\phi}_n^T(\mathbf{x}_i) = \{\beta_i \boldsymbol{\phi}_{n,i} \boldsymbol{\phi}_{n,i}^T\}_i \qquad (6)$$

is the Fisher information matrix.

We now address learning the optimal c_i. An *n*th order KMP employs *n* basis functions. According to the definition in (3), the (*n*+1)th order KMP is inductively written as

$$f_{n+1}(\mathbf{x}) = \mathbf{w}_{n+1}^T \boldsymbol{\phi}_{n+1}(\mathbf{x}) \qquad (7)$$

where

$$\boldsymbol{\phi}_{n+1}(\cdot) = [1, K(\mathbf{c}_1, \cdot), K(\mathbf{c}_2, \cdot), \ldots, K(\mathbf{c}_n, \cdot), K(\mathbf{c}_{n+1}, \cdot)]^T = \begin{bmatrix} \boldsymbol{\phi}_n(\cdot) \\ \phi_{n+1}(\cdot) \end{bmatrix} \qquad (8)$$

with $\phi_{n+1}(.) = K(\mathbf{c}_{n+1}, \cdot)$ a new basis function centered at $\mathbf{c}_{n+1}$. The weighted sum of squared errors of the (n+1) th order KMP is

$$e_{n+1} = (1 \big/ \sum_{i=1}^{N} \beta_i) \sum_{i=1}^{N} \beta_i [y_i - f_{n+1}(\mathbf{x}_i)]^2 \tag{9}$$

Assuming the basis functions in $\boldsymbol{\phi}_{n+1}$ are all known, then from (5)

$$\mathbf{w}_{n+1} = \mathbf{M}_{n+1}^{-1} \{\beta_i \boldsymbol{\phi}_{n+1,i} y_i\}_i \tag{10}$$

minimizes (9), where the Fisher information matrix $\mathbf{M}_{n+1}$ is given as

$$\mathbf{M}_{n+1} = \{\beta_i^2 \boldsymbol{\phi}_{n+1,i} \boldsymbol{\phi}_{n+1,i}^T\}_i \tag{11}$$

One may show that $\mathbf{w}_{n+1}$, and e_{n+1} are respectively related to $\mathbf{w}_n$ and e_n as

$$\mathbf{w}_{n+1} = \begin{bmatrix} \mathbf{w}_n + \mathbf{M}_n^{-1} \{\beta_i \boldsymbol{\phi}_{n,i} \phi_{n+1,i}\}_i b^{-1} [\{\beta_i \boldsymbol{\phi}_{n,i}^T \phi_{n+1,i}\}_i \mathbf{w}_n - \{\beta_i \phi_{n+1,i} y_i\}_i] \\ -b^{-1} \{\beta_i \boldsymbol{\phi}_{n,i}^T \phi_{n+1,i}\}_i \mathbf{w}_n + b^{-1} \{\beta_i \phi_{n+1,i} y_i\}_i \end{bmatrix} \tag{12A}$$

$$e_{n+1} = e_n - \delta e(K, \mathbf{c}_{n+1}) \tag{12B}$$

where

$$\delta e(K, \mathbf{c}_{n+1}) = (1 \big/ \sum_{i=1}^{N} \beta_i) b^{-1} [\{\beta_i \boldsymbol{\phi}_{n,i}^T \phi_{n+1,i}\}_i \mathbf{w}_n - \{\beta_i \phi_{n+1,i} y_i\}_i]^2 \tag{13}$$

and

$$b = \{\beta_i \phi_{n+1,i}^2\}_i - \{\beta_i \phi_{n+1,i} \boldsymbol{\phi}_{n,i}^T\}_i \mathbf{M}_n^{-1} \{\beta_i \boldsymbol{\phi}_{n,i} \phi_{n+1,i}\}_i \tag{14}$$

with $\phi_{n+1,i} = K(\mathbf{c}_{n+1}, \mathbf{x}_i)$. It may be stressed that formulae (13)-(14) provide a crucial method for reducing the computational complexity. These techniques enable very fast design of kernel machines to be performed, even on large datasets.

With sufficient training data points, we can always make $\mathbf{M}_{n+1}$ positive definite. Then $\mathbf{M}_{n+1}^{-1}$ is also positive definite and it holds $b^{-1} > 0$, which guarantees $\delta e(K, \mathbf{c}_{n+1})$ is always greater than zero. Therefore, from (12B), $e_{n+1} < e_n$, which means appending a new basis function to the KMP generally leads to decrease of the representation error *on the training sample;* the effect on generalization is more complex and has been described in the previous section.

Since $\delta e(K, \mathbf{c}_{n+1})$ is dependent on the center $\mathbf{c}_{n+1}$ of the new basis function, we obtain different values of $\delta e(K, \mathbf{c}_{n+1})$ by selecting different $\mathbf{c}_{n+1}$. If we confine $\mathbf{c}_{n+1}$ to be selected from the training data, we may conduct a "greedy" search in the training set but with the previously selected data excluded to avoid repetition, selecting the datum that maximizes (13). Formally, we have

$$\mathbf{c}_{n+1} = \mathbf{x}_{i_{n+1}} = \arg\max_{\substack{k \neq i_1,\ldots,i_n \\ 1 \leq k \leq N}} \delta e(K, \mathbf{x}_k) \tag{15}$$

After $\mathbf{c}_{n+1}$ is determined, we update the weights using (12A) and the Fisher information matrix.

From (13) $\delta e(K, \mathbf{c}_{n+1})$ depends on the functional form of the kernel $K(\cdot,\cdot)$ as well as on support samples $\mathbf{c}_{n+1}$. This allows us to optimize the kernel to gain further error reduction. A simple approach to take is to first conduct a "greedy" search of $\mathbf{c}_{n+1}$ in the training set, for a fixed kernel, and then fix $\mathbf{c}_{n+1}$ and optimize the parameters of the kernel. For radial basis function (RBF) kernels, the only parameter other than $\mathbf{c}_{n+1}$ is the kernel width, thus optimization of RBF kernels with $\mathbf{c}_{n+1}$ fixed is a one-dimensional search for the kernel width. It is also possible to optimize $\mathbf{c}_{n+1}$ and the kernel width simultaneously, but then $\mathbf{c}_{n+1}$ is treated as a free parameter and is no longer confined to the training set. Another possibility is optimization over kernels of different functional forms, which offers greater diversity of the basis functions available to the KMP.

B. Life Long Learning

Assume that an MCM sensor system has previously performed *M*-1 sensing tasks, with each task characterized by a particular environment, mines and clutter. Now consider a new sensing task, for a total of *M* tasks. Assume that the *m*th task is characterized by N_m labeled signatures, *i.e.*, the labeled data for task *m* are $D_m = \{(\boldsymbol{x}_{nm}, y_{nm}):\ n = 1,\ldots,N_m\}$, where $\boldsymbol{x}_{nm}$ is a *d*-dimensional real feature vector and $y_{nm} \in \{-1,\ 1\}$ is the associated label. Our objective is to design a classifier for the new task *M* while leveraging the related information available from the previous *M*-1 tasks. The algorithm discussed below automatically determines which of the *M*-1 previous tasks are relevant for learning an algorithm for task *M*, while minimizing the importance of the tasks that are not relevant.

The learning algorithm discussed here simultaneously designs a classifier for each of the *M* tasks, and in each case information (data) from the other tasks is shared if deemed relevant. Consequently there are two important applications of the algorithm:

- *Life-long learning*, in which the algorithm trained for a new task *M* is placed within the context of all previous *M*-1 tasks (*i.e.*, placed within context of historical data). In this manner the algorithm designed for a new task exploits all relevant information from previous tasks. This has the important property of reducing the quantity of labeled data required for each of the individual tasks, since data are shared among all tasks.

- For multiple MCM platforms, a networked suite of distributed sensors observe different portions of the environment. Assume *M* sensor platforms collect data. The processing of these data may be viewed as *M* tasks, and it is desirable to integrate the execution of these multiple tasks, yielding *multi-task learning*. In multi-task learning, when analyzing a particular task, data from the other *M*-1 tasks are appropriately exploited. Consequently, in the context of a multi-platform UUV solution, the data from each platform is viewed as a task, and the multi-task learning algorithm optimally shares information across tasks.

A nonparametric Bayesian model is considered for jointly learning multiple classifiers, each corresponding to a task, with an associated dataset. In particular, we employ the Dirichlet Process Mixture (DPM) as the common prior on the model parameters of the tasks. The model automatically identifies task clusters via Bayesian inference. The main advantage of a nonparametric model is that *it makes no assumptions regarding the underlying distributions*, and therefore it provides a richer and more flexible representation than its parametric counterparts.

Recall that the labeled data for task *m* are represented as $D_m = \{(\boldsymbol{x}_{nm}, y_{nm}): \; n = 1,..., N_m\}$, and our objective is to learn classifiers for each of the

M tasks, by simultaneously sharing information (data) deemed relevant by the multi-task-learning algorithm. For each task m the conditional probability of label y_{mn} given $\boldsymbol{x}_{mn}$ is modeled via logistic regression

$$p(y_{nm}|\boldsymbol{w}_m, \boldsymbol{x}_{nm}) = \sigma(y_{nm}\boldsymbol{w}_m^T\boldsymbol{x}_{nm}) \tag{16}$$

where $\boldsymbol{w}_m$ parameterizes the classifier for task m, and $\sigma(x) = \exp(x)/[1+\exp(x)]$. The goal is to learn $\{\boldsymbol{w}_m\}_{m=1,M}$ *jointly* such that the resulting classifiers can accurately predict class labels for new test samples. The hierarchical model of $\{\boldsymbol{w}_m\}_{m=1,M}$ is specified as

$$\boldsymbol{w}_m|\boldsymbol{\theta}_m \sim F(\boldsymbol{w}_m|\boldsymbol{\theta}_m), \quad \boldsymbol{\theta}_m|G \sim G, \quad G \sim DP(\alpha, G_o) \tag{17}$$

where $DP(\alpha, G_o)$ is a Dirichlet process with precision parameter α and base distribution G_o. The Dirichlet process is used to account for the uncertainty of G. Using Sethuraman's stick-breaking representation, we can write

$$G = \sum_{k=1}^{\infty}\pi_k(\boldsymbol{v})\delta_{\theta_k^*}, \qquad \pi_k(\boldsymbol{v}) = v_k\prod_{i=1}^{k-1}(1-v_i), \qquad v_k \sim Beta(1,\alpha), \qquad \theta_k^* \sim G_o. \tag{18}$$

Using (17) and (18) we have

$$\boldsymbol{w}_m|\{\boldsymbol{\theta}_k^*\} \sim \prod_{k=1}^{\infty}(F(\boldsymbol{w}_m|\boldsymbol{\theta}_k^*))^{c_{m,k}} \tag{19}$$

where c_{mk} is a cluster membership indicator defined as $c_{mk} = 1$ if $\boldsymbol{w}_m$ belongs to cluster k, and $c_{mk} = 0$ otherwise. The clustering structure of $\boldsymbol{w}_m$ represents *relatedness among tasks*.

The use of Dirichlet processes in Bayesian inference represents the state of the art in Bayesian analysis, providing flexibility and generality not available in traditional approaches. However, an important challenge that must be addressed is computation of the integrals required for inference. To address this challenge we utilize variational Bayes (VB) inference, which we discuss next.

Assume the model parameters of interest are represented by the vector $\boldsymbol{\theta}$; we hope to learn these parameters based on observed data *D*. For density function estimation $\boldsymbol{\theta}$ may represent the parameters of a Gaussian mixture model (GMM), while in a classification problem $\boldsymbol{\theta}$ may represent the weights *w* in the incomplete-data logistic-regression classifier.

Our objective is to obtain the posterior probability distribution of the hidden variables $\boldsymbol{\theta}$ based on a set of observed variables *D* (for GMM design the data *D* is unlabeled, while for the logistic-regression classifier *D* is labeled or imperfectly labeled data). Since an exact inference of the hidden variables $\boldsymbol{\theta}$ based on the observed variables *D* is intractable for all but the simplest model structures, our goal is to find a tractable variational distribution $Q(\boldsymbol{\theta})$ that closely approximates the true posterior distribution $p(\boldsymbol{\theta}|\boldsymbol{D})$.

Let *p*(*D*) denote the marginal probability of the observed data *D*. The log-marginal can be written as

$$\ln p(\boldsymbol{D}) = L(Q) + KL(Q\|P') \tag{20}$$

where

$$L(Q) = \sum_{\theta} Q(\boldsymbol{\theta}) \ln \frac{p(\boldsymbol{D}|\boldsymbol{\theta})\, p(\boldsymbol{\theta})}{Q(\boldsymbol{\theta})} \tag{21}$$

and

$$KL(Q\|P') = \sum_{\theta} Q(\boldsymbol{\theta}) \ln \frac{p(\boldsymbol{\theta}|\boldsymbol{D})}{Q(\boldsymbol{\theta})} \tag{22}$$

with $P' = p(\boldsymbol{\theta}|\boldsymbol{D})$. The summations in (21) and (22) are replaced by integrals if the hidden variables $\boldsymbol{\theta}$ are continuous.

Note that the above expression is true for any approximating variational distribution $Q(\boldsymbol{\theta})$. The term $KL(Q\|P')$ represents the Kullback-Leibler (KL) divergence between the true posterior $p(\boldsymbol{\theta}|\boldsymbol{D})$ and its variational approximation $Q(\boldsymbol{\theta})$. Our objective is to optimize $Q(\boldsymbol{\theta})$ to minimize the KL divergence between $Q(\boldsymbol{\theta})$ and $p(\boldsymbol{\theta}|\boldsymbol{D})$.

However, since the posterior density function $p(\boldsymbol{\theta}|\boldsymbol{D})$ is known, and is the subject of this analysis, the KL divergence in (7) cannot be evaluated. However, since $KL(Q\|P')$ is always non-negative, the term $L(Q)$ forms a lower bound of the log-marginal, $\ln p(\boldsymbol{\theta})$. Consequently, minimization of $KL(Q\|P')$ with respect to Q is equivalent to maximization of $L(Q)$ since the left hand side $\ln p(D)$ is independent of the variational distribution Q. All of the terms in (6) can be evaluated, and therefore the variational Bayes (VB) approximation to $p(\boldsymbol{\theta}|\boldsymbol{D})$ reduces to attempting to determine the $Q(\boldsymbol{\theta})$ that maximizes the variational expression $L(Q)$.

For the sake of tractability, we assume that the hidden variables are independent of each other, meaning $Q(\boldsymbol{\theta})$ may be written in a factorized form as $Q(\boldsymbol{\theta}) = \prod_i Q_i(\theta_i)$, where $\{\theta_i\}$ is the set of disjoint hidden variables indexed by i constituting $\boldsymbol{\theta}$. In variational inference we optimize the factors of the variational distribution one at a time, cycling sequentially through all factors. We accomplish this by separating out the terms involving a factor $Q_i(\theta_i)$ (approximating the distribution for hidden variable θ_i). We can therefore maximize the lower bound $L(Q)$ with respect to a single factor Q_i (assuming all $Q_{j,j\neq i}$ are temporarily fixed), and then cycling through each hidden variable θ_i in turn replacing the current distribution $Q_i(\theta_i)$ with a revised estimate $Q_i^*(\theta_i)$.

This iterative VB analysis can be performed efficiently if each $Q_i(\theta_i)$ is conjugate to the likelihood function with all $\theta_{j,j\neq i}$ equal to a constant. Specifically, this conjugacy property allows the update equations to be performed analytically, thereby yielding a VB algorithm with computational speed commensurate with the widely used EM algorithm employed in ML point estimates of the parameters $\boldsymbol{\theta}$. Fortunately, many models have a structure that is directly amenable to appropriate conjugate priors.

C. *In Situ* Learning

Assume that the labeled data with which classifier design may be performed is denoted D_L, and the unlabeled data at the new site of interest is represented as D_U. We consider a nonlinear classifier based on a set of N_B basis functions $\{\boldsymbol{b}_n\}_{n=1,N_B}$, where the basis functions are determined using the techniques discussed above in the context of KMP. We again utilize the kernel-based function

$$f(\boldsymbol{x};\boldsymbol{w}) = \sum_{n=1}^{N_B} w_n K(\boldsymbol{x},\boldsymbol{b}_n) + w_0 = \boldsymbol{w}^T \boldsymbol{K}(\boldsymbol{x}) \tag{23}$$

and the probability that x is associated with label y=1 is expressed as

$$p(y=1|\boldsymbol{x},\boldsymbol{w}) = \sigma[f(\boldsymbol{x};\boldsymbol{w})] = 1/\{1+\exp[f(\boldsymbol{x};\boldsymbol{w})]\} \tag{24}$$

with $p(y=-1|\boldsymbol{x},\boldsymbol{w}) = 1 - p(y=1|\boldsymbol{x},\boldsymbol{w})$. In (23) w is an N_B+1 dimensional vector composed of the weights $\{w_0, w_1,..., w_{N_B}\}^T$, with the N_B+1 dimensional vector $K(x)$ defined in terms of the components $K(x,x_n)$. The function $K(x,x_n)$ is a general kernel defining the similarity of feature vectors x and x_n. The radial basis function, $K(\boldsymbol{x},\boldsymbol{x}_n) = \exp(\frac{1}{2\alpha^2}\|\boldsymbol{x} - \boldsymbol{x}_n\|^2)/\sqrt{2\pi\alpha^2}$, represents one class of kernels that may be employed. It is important to emphasize that the classifier in (23) and (24) yields a probabilistic measure as to the confidence that a given feature vector x is associated with a given label y, thereby presenting the decision maker with a level of algorithmic confidence.

Assume the kernel-based classifier in (23) and (24) is trained using the N_L labeled signatures in D_L. As a consequence of this training we yield a posterior estimate of the weights given the training data, $p(\boldsymbol{w}|\boldsymbol{D}_L)$. We may compute the information accrued in w via the data D_L via the N_B+1 × N_B+1 dimensional Fisher information matrix, with ij element

$$F_{ij} = -E[\frac{\partial}{\partial w_i}\frac{\partial}{\partial w_j}\log p(\mathbf{w}|\mathbf{D}_1)] \tag{25}$$

As is well known, the Cramer-Rao bound is defined by the inverse of the Fisher information matrix, this defining the minimum variance with which one may estimate the weights w given the finite labeled data D_L. To a good approximation the Fisher information matrix based on D_L may be expressed as

$$\boldsymbol{F}(\boldsymbol{D}_L) = \sum_{n=1}^{N_L} \boldsymbol{K}(\boldsymbol{x}_n)^T \boldsymbol{K}(\boldsymbol{x}_n)\sigma[f(\boldsymbol{x}_n,\boldsymbol{w})]\sigma[-f(\boldsymbol{x}_n,\boldsymbol{w})] \tag{26}$$

We may now quantify the maximum information that may be added if we acquire a label for that member of the unlabeled data D_U for which label acquisition would be most informative

$$\delta \equiv \max_{\boldsymbol{x}\in\boldsymbol{D}_U} trace\ \{\boldsymbol{F}(\boldsymbol{D}_L) + \boldsymbol{K}(\boldsymbol{x})^T \boldsymbol{K}(\boldsymbol{x})\sigma[f(\boldsymbol{x}_n,\boldsymbol{w})]\sigma[-f(\boldsymbol{x}_n,\boldsymbol{w})]\} \tag{27}$$

In (27) we have employed the trace, but any matrix measure may be used, such as the determinant. In any case, (27) quantifies the information content that may be accrued with regard to estimating the classifier weights w, based upon acquiring the label for the single most informative member of the new (testing) data D_U.

Using the measure δ above, one may quantify which element of $\boldsymbol{x}\in\boldsymbol{D}_U$ would be most informative to classifier design if it could be employed within the training phase. To use this $\boldsymbol{x}\in\boldsymbol{D}_U$ while training, the associated label y is required. Hence δ provides feedback as to which element of D_U is most desirable for label acquisition. This label may then be acquired via personnel, by near-range possibly unmanned sensors, or via an analyst.

This is termed *in situ* learning because the algorithm automatically infers which unlabeled signatures from the site of interest would be most informative to classifier design if the associated labels could be acquired. This algorithm may be applied for cases in which there is no or little pre-existing labeled data sets for training.

D. Feature Extraction

The implementation of identification algorithms is predicated on extracting features from the target strength data collected from unknown objects that are in an area to be cleared of mines. Save the necessity of collecting raw data containing exploitable differences between mines and clutter, feature extraction is paramount. Our overall design clearly delineates feature extraction from identification using the extracted features. The unknown nature of the clutter in various environments we plan on measuring in the near future precludes discussion of the final features that will be employed in the operational system. However, the features used today and the methodology used to select them are discussed in turn. Following these discussions the issue, which was deferred earlier in the document, of how to combine the advantages *discriminative* approach with the sequential nature of the data is discussed.

Feature extraction converts the structural acoustic information contained in the scattered, multi-aspect, acoustic signals from targets in the low frequency band into a form that can be utilized by the identification algorithms. Signal processing techniques are used to produce multi-aspect target strength (either in the time of frequency domain) from the scattered signals. Features are then extracted from the target strength to implement mine identification.

To date, four different feature sets were explored when analyzing the identification performance of the system: normalized energy in sub-bands, wavelet moments, relaxed matching pursuits, and central moments. In general, the normalized energy features worked as well as or better than the other feature sets for the most challenging identification scenarios, and the results using this feature set are presented in previous reports about the program. Thirty six equally distributed frequency bands are used for the performance estimation of the system. The normalized energy features are the ratio of the energy in a frequency band normalized by the total energy in the signal:

$X = \frac{\int_{f_1}^{f_2} |P(f)|^2 \, df}{\int_{0}^{\infty} |P(f)|^2 \, df}$, where $P(f)$ is the signal's spectrum, and f_1 and f_2 define the frequency band.

The normalized energy features have been robust to the channel variations do to the changing depth to range ratio encountered thus far. Intuitively, this is due to the fact that the spectral beating, controlled by the arrival times that change when the depth to range ratio changes, is averaged out by the integral in the numerator of the expression for the normalized energy features..

We have considered two methodologies for feature selection. The more-sophisticated approach is called joint feature and classifier optimization (JCFO), it representing an extension of the aforementioned kernel-based classifiers. Specifically, we employ an augmented kernel representation of the following form

$$f(\boldsymbol{x};\boldsymbol{w},\boldsymbol{\theta}) = \sum_{n=1}^{N_B} w_n K(\boldsymbol{x},\boldsymbol{b}_n;\boldsymbol{\theta}) + w_0 = \boldsymbol{w}^T \boldsymbol{K}(\boldsymbol{x};\boldsymbol{\theta}) \tag{28}$$

where a new vector $\boldsymbol{\theta}$ is introduced, this of dimension *d*, corresponding to the dimensionality of the feature vector *x*. The kernel is represented as

$$K(\boldsymbol{x},\boldsymbol{b}_n;\boldsymbol{\theta}) = \exp[-\sum_{i=1}^{d} \theta_i (x_i - b_{n,i})^2] \tag{29}$$

where x_i is the *i*th component of the vector *x*, and $b_{n,i}$ is the *i*th component of the vector b_n. The scalar θ_i weights the importance of the *i*th feature in the classifier. In the JCFO algorithm the goal is to learn the vectors *w* and $\boldsymbol{\theta}$. A sparseness prior (regularizer) is placed on both of these parameters, such that in the final classifier most components of *w* and $\boldsymbol{\theta}$ are zero or near-zero. We thereby simultaneously learn which basis vectors are most relevant for classifier design (those with non-zero corresponding components in *w*), as well as the most-relevant features (those with non-zero components in $\boldsymbol{\theta}$). This simultaneous learning of the classifier design and the associated features plays an important role in JCFO performance, since the optimal features are a function of the specific classifier employed, and *vice versa*.

While the JCFO algorithm represents an excellent tool for classifier design and feature selection, the fact that we must solve for two vectors, *w* and $\boldsymbol{\theta}$, makes the algorithm relatively slow for large data sets. We therefore also utilize the simpler approach of designing a classifier using the functional

$$f(\boldsymbol{x};\boldsymbol{w}) = \sum_{n=1}^{d} w_n x_n + w_0 = \boldsymbol{w}^T \boldsymbol{x} \tag{30}$$

where now the dimensionality of the vector *w* is equal to *d*, the number of features. We place a sparseness prior on *w*, and thereby select the most relevant features when performing classifier design. We have found this algorithm to often serve as an excellent tool for designing a classifier and selecting features, and the computational speed of this approach is typically quite fast. In many practical applications one may utilize (30) to determine an initial weighting on the importance of features, with the insight so learned used to initialize the JCFO algorithm, yielding improved JCFO convergence properties.